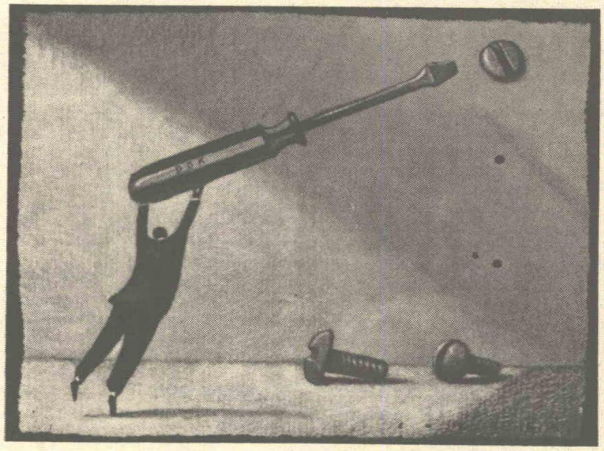
มนตรี กาญจนะเดชะ

SOFT MAIL

PKILL
โปรแกรมฆ่าโปรแกรม

พบกับคุณสมบัติของโปรแกรมฆ่าโปรแกรมที่มีประโยชน์มากมายจนคุณคาดไม่ถึง



ก่อนอื่นผู้เขียนขอเล่าถึงที่มาของโปรแกรม PKILL นี้สักเล็กน้อย เมื่อ 4 ปีที่แล้วขณะที่ผมกำลังศึกษาอยู่ มีเพื่อนของผมคนหนึ่งมาขอยืมเครื่องคอมพิวเตอร์เพื่อใช้สำหรับพิมพ์รายงาน (ใช้เวิร์ดจุพา) หลังจากนั่งคีย์รายงานอยู่จนถึงเวลาประมาณตี 3 เพื่อนผมก็วิ่งมาหาผมบอกว่า เครื่องแฮงไปแล้ว ผมจึงไปดูอาการและสืบหาสาเหตุ หลังจากที่ผมได้สอบถามเพื่อนผม จึงพบว่าสาเหตุที่ทำให้เครื่องแฮงมาจากการที่ผมได้ตั้งชื่อไฟล์โดยมีเครื่องหมายสเปซบาร์แทรกอยู่ ทำให้การทำงานของเวิร์ดจุพาผิดพลาด (ขณะนั้นเวิร์ดจุพายังไม่สมบูรณ์เหมือนปัจจุบัน) เพื่อนผมจึงต้องนั่งคีย์รายงานใหม่ทั้งหมด

ได้ความคิดใหม่

จากเหตุการณ์นั้นทำให้ผมกลับมาคิดหาวิธีที่ทำให้เครื่องที่แฮงนั้นสามารถกลับมาทำงานได้เหมือนเดิม เพราะในกรณีของเพื่อนผม ถ้าผมสามารถกลับมาที่ดอสได้อีก ผมก็สามารถหาข้อมูลที่ยังไม่ได้ถูกเก็บลงดิสค์ (ยังคงอยู่ในหน่วยความจำ) นำมาเก็บลงดิสค์เอง

หลังจากที่ได้ศึกษาหาข้อมูลเพิ่มเติมระยะหนึ่ง ผมก็ได้ใช้ภาษาแอสเซมบลีเขียนโปรแกรม PKILL.COM ขึ้นมา ซึ่ง PKILL ย่อมาจาก Process Killer หรือ Program Killer ก็ได้

ขั้นตอนการทำงานโปรแกรม

จากเวอร์ชันแรกของ PKILL.COM จนถึงปัจจุบันก็มีการปรับปรุงมาเรื่อย ๆ ซึ่งพอจะสรุปขั้นตอนการทำงานของโปรแกรมได้ดังนี้

1. ตรวจสอบการติดตั้งตัวเองซ้ำ (PKILL เป็นโปรแกรมแบบ TSR)
2. เก็บ Interrupt Vector ที่จำเป็น
3. เก็บโหมดจอภาพปัจจุบัน และโหมดการทำงานปัจจุบันของ 8259
4. ย้ายอินเตอร์รัพต์ 9 ให้ชี้มาที่ตัวเอง
5. แสดงข้อความว่าได้ถูกติดตั้งแล้ว และฝังตัวอยู่ในหน่วยความจำ
6. ตรวจสอบการกดคีย์บอร์ด ถ้าคีย์ที่กดเป็นแอคตีฟคีย์ก็จะทำงาน
7. ถ้าแอคตีฟคีย์เป็น Ctrl-Alt-Right Shift ซึ่งหมายถึง คำสั่งให้ฆ่าโปรแกรมที่แฮงออก โดย PKILL จะฆ่าโปรแกรมทุกโปรแกรมที่รันหลัง PKILL การฆ่าจะทำการคืนหน่วยความจำของโปรแกรมที่ถูกฆ่าให้ดอส และจะทำการเซตเครื่องให้เป็นเหมือนกับตอนก่อนที่จะมีการติดตั้ง PKILL โดยใช้ข้อมูลที่ได้เก็บไว้ในข้อ 2 และข้อ 3 จากนั้นสั่งปิดเสียงจากลำโพง เคลียจอภาพ และแสดงข้อความว่า PKILL ได้ทำงานแล้ว และแสดงว่ามีการฆ่าโปรแกรมใดไปบ้าง ขั้นตอนสุดท้ายจะเรียกใช้งานฟังก์ชัน 4CH ของอินเตอร์รัพต์ 21H เพื่อคืนการทำงานให้ดอส
8. ถ้าแอคตีฟคีย์เป็น Ctrl-Alt-Left Shift-Right Shift ซึ่งหมายถึง ต้องถอนตัวเองออกจากหน่วยความจำ PKILL ก็จะทำงานเหมือนกับข้อ 7 แต่จะฆ่าตัวเองด้วย

เนื่องจาก PKILL ใช้ฮาร์ดแวร์อินเตอร์รัพต์ในการกระตุ้นการทำงาน (INT 9) ดังนั้น PKILL สามารถถูกเรียกใช้ให้ทำงานได้ตลอดเวลา ยกเว้นในกรณีที่ตัวโปรแกรม PKILL ที่ฝัง

อยู่ในหน่วยความจำถูกทำลาย, Interrupt vector ถูกทำลาย หรือเมื่อมีการโปรแกรมฮาร์ดแวร์ให้ไม่มีการเกิดการอินเตอร์-รัพต์เมื่อมีการกดคีย์บอร์ด โปรแกรม PKILL ก็ไม่สามารถทำงานได้ ดังนั้นโปรแกรม PKILL จึงไม่สามารถช่วยแก้เครื่องแฮงได้ในทุกกรณี แต่จะช่วยได้ในกรณีที่ไม่หนักหนาเกินไปนัก

คุณสมบัติของโปรแกรม

วัตถุประสงค์ในการสร้างโปรแกรม PKILL เริ่มแรกก็เพียงเพื่อต้องการใช้แก้เครื่องแฮงเท่านั้น แต่หลังจากที่ได้แจกจ่ายให้เพื่อน ๆ ได้ใช้งานและใช้เองมาจนถึงปัจจุบัน ปรากฏว่า โปรแกรมนี้มีประโยชน์มากมายอย่างคาดไม่ถึง ซึ่งพอสรุปได้ดังนี้

1. ใช้แก้เครื่องแฮง

2. ใช้ออกจากโปรแกรมใด ๆ ในกรณีที่ไม่รู้ว่ามีวิธีออกอย่างไร

3. ใช้ออกจากโปรแกรมที่ไม่มีทางออก เช่น โปรแกรมเกมรุ่นเก่า ๆ ซึ่งถ้าต้องการเลิกเล่นก็ต้องรีเซตเครื่องอย่างเดียว เช่น โปรแกรม PCMAN เป็นต้น

4. ใช้สำหรับนักเรียนนักศึกษาผู้เริ่มหัดเขียนโปรแกรมภาษาแอสเซมบลี เนื่องจากแอสเซมบลีเป็นภาษาที่ละเอียดอ่อน โอกาสที่จะเขียนโปรแกรมผิด ๆ แล้วทำให้เครื่องแฮงจึงมีสูง

5. ใช้ฆ่าโปรแกรมแบบ TSR ทั้งหลาย

6. ใช้สำหรับผู้เริ่มต้นหัดเขียนโปรแกรมแบบ TSR ผู้ที่เคยเขียนโปรแกรมประเภทนี้คงจะทราบดีว่า ในการทดสอบโปรแกรมจะต้องมีการรีเซตเครื่องบ่อยมาก

7. ใช้ฆ่าไวรัส ถ้าก่อนการติดตั้ง PKILL ไม่มีไวรัสฝังตัวอยู่ในหน่วยความจำ หลังจากติดตั้ง PKILL แล้วพบว่ามีไวรัสแอคตีฟอยู่ในหน่วยความจำ ท่านก็สามารถกดแอคตีฟคีย์เรียก PKILL ให้ทำงานเพื่อฆ่าไวรัส โดยใช้ความสามารถของ PKILL ในการเซต Interrupt Vector ต่าง ๆ คืน และการคืนหน่วยความจำให้ดอสได้

8. ใช้แก้เกมที่มี Password (วิธีนี้พบโดยเพื่อนของผม) วิธีการคือรันเกมผ่านโปรแกรม Debugger เมื่อโปรแกรมเกมถาม Password ก็ทำการกดแอคตีฟคีย์ โปรแกรมเกมจะถูก Terminate แล้วจะกลับมาทำงานที่ Debugger ต่อ (จริง ๆ แล้วควรจะกลับไปที่ดอส ผมก็ไม่เข้าใจเหมือนกันว่าเพราะเหตุใด) จากนั้นก็ใช้ความรู้ภาษาแอสเซมบลีเพื่อแก้ไขโปรแกรมเกม (ยังคงอยู่ในหน่วยความจำ) ในส่วนที่จัดการเกี่ยวกับ Password

การติดตั้งโปรแกรม PKILL ควรจะพิจารณาลำดับการฝังตัวลงหน่วยความจำด้วย เพราะเมื่อเวลาแอคตีฟ PKILL จะฆ่าโปรแกรมที่ฝังตัวหลังจากตัวมันทั้งหมด ดังนั้นโปรแกรมที่ต้องการให้ทำงานตลอดเวลา เช่น โปรแกรมไดรเวอร์ของเม้าส์ หรือโปรแกรมจำพวกดิสค์แคชควรจะถูกติดตั้งไว้ก่อน

ก่อนจบ

จากการใช้งานโปรแกรม PKILL ที่ผ่านมายังไม่มีปัญหาในการทำงานร่วมกับโปรแกรมอื่น ๆ รวมทั้งไมโครซอฟต์ ถ้าท่านผู้อ่านมีข้อสงสัย ข้อแนะนำต่าง ๆ ก็สามารถติดต่อกับทางกองบรรณาธิการได้ครับ

ลิสติงที่ 1 โปรแกรม PKILL.ASM

```
; — Last Update  JULY 13 1991 —

ACT_KEY         EQU     0DH             ; CTRL-ALT-RIGHT SHIFT Key
                                        ; KILL PROCESS ACTIVE KEY RM_KEY
EQU             0FH                     ; CTRL-ALT-LEFT-RIGHT Key
                                        ; DEINSTALL PKILL.COM ACTIVE KEY
FALSE           EQU     0
TRUE            EQU     1


CODE            SEGMENT
                ASSUME CS:CODE,DS:CODE
                ORG     100H

MAIN:           JMP     INSTALL

MARKER1                 DW      'PO'    ; First Marker
OLD_09_VEC              DD      ?       ; Old INT 9 vector
VDO_MODE                DB      ?       ; Current video mode buffer
VDO_PAGE                DB      ?       ; Current video page buffer
OLD_8259                DB      ?       ;
OLD_8255                DB      ?       ;
KEY_IN                  DB      ?       ; Buffer for active key checking
KEY_TMP                 DB      ?

INT_09H         PROC    FAR
                STI                     ; Turn Interrupt on.
                PUSHF                   ; PUSH Flags
                CALL    CS:OLD_09_VEC   ; Does INT  9 , read a keyboard.
                PUSH    AX              ;
                MOV     AH,2            ;
                INT     16H             ; Get keyboard shift status
                AND     AL,0FH          ; Delete high nibble
                MOV     CS:KEY_IN,AL
                POP     AX

                CMP     BYTE PTR CS:KEY_IN,ACT_KEY      ; Is my actived key?
                JZ      MY_ACT_KEY                      ; Yes, kill all programs.
                CMP     BYTE PTR CS:KEY_IN,RM_KEY       ; Is deinstall key?
                JZ      MY_ACT_KEY                      ; Yes, REMOVE PKILL.COM
EXIT_@@:        IRET                                    ; No, exit interrupt routine.

MY_ACT_KEY:
                PUSH    AX
                MOV     AL,CS:KEY_IN
                MOV     CS:KEY_TMP,AL
                POP     AX
                PUSH    AX              ;++++++++++++++++
                PUSH    BX              ;
                PUSH    CX              ;
                PUSH    DX              ; Save registers.
                PUSH    SI              ;
                PUSH    DI              ;
                PUSH    DS              ;
                PUSH    ES              ;++++++++++++++++
```

ลิสติงที่ 1 (ต่อ)

```
                CLI

                PUSH    CS
                POP     DS
; ------ Set Display to default display mode ---

                MOV     AL,VDO_MODE     ; Default video mode in AL
                XOR     AH,AH           ; Function 0
                INT     10H             ; Call bios function
                MOV     AL,VDO_PAGE     ; Default video page in AL
                MOV     AH,5            ; Set video page function
                INT     10H             ; Call bios function

; ------ Clear display screen and move cursor to coordinate (0,0) ---

                MOV     AX,600H         ; AH = Function No. 6, AL = 0
                XOR     CX,CX           ; Begin block = (0,0)
                MOV     DX,284FH        ; End block = (24,79)
                MOV     BH,7            ; Fill with normal attribute
                INT     10H             ; Actived clear screen
                MOV     AH,2            ; Function 2, set cursor position
                XOR     DX,DX           ; Set to coordinate (0,0)
                MOV     BH,VDO_PAGE     ; BH = default video page
                INT     10H             ; Actived set cursor

;-------- Display message --------

                PUSH    CS
                POP     ES
                MOV     SI,OFFSET WAKE_MSG
                CALL    WRITE_CH

                MOV     DX,CS           ; DS=CODE SEGMENT
                DEC     DX              ; DX = This program MCB's segment
NEXT_MCB:
                MOV     ES,DX           ; DS=>Segment of MCB
                MOV     AX,WORD PTR ES:[3]  ; Get length in paragraph
                ADD     AX,DX           ; Find next MCB's Segment
                INC     AX              ; AX = Next MCB's segment
                MOV     DX,AX           ; And save it in DX
                MOV     ES,DX           ; ES=MCB SEGMENT
                CMP     BYTE PTR ES:[0],'Z'  ; Is last MCB ?
                JZ      END_ALLOC       ; Yes, end allocation
                MOV     AX,ES:[1]       ; GET ID
                PUSH    ES              ; save es
                PUSH    CS
                POP     ES
                MOV     SI,OFFSET MESSAGE0
                CALL    WRITE_CH

;--- CHECK ENVIRONMENT ---
                MOV     ES,DX
                CMP     ES:[10H],'OC'
                JNZ     NOT_ENVI
                CMP     ES:[10H+2],'SM'
                JNZ     NOT_ENVI
                CMP     ES:[10H+4],'EP'
                JNZ     NOT_ENVI
                CMP     ES:[10H+6],'=C'
                JNZ     NOT_ENVI

;-- DISPLAY ENVIRONMENT REMARK ---

                PUSH    CS
                POP     ES
                MOV     SI,OFFSET ENVI_MSG
                CALL    WRITE_CH

;--- REMARK ** MAKE SURE THAT AX = PROGRAM CODE SEGMENT ---

NOT_ENVI:
                CALL    SEARCH_PROG     ; SEARCH PROGRAM NAME
                CALL    WRITE_CH        ; WRITE PROGRAM NAME
                PUSH    CS
                POP     ES              ; ES=CS
                MOV     SI,OFFSET CR_LF ; next line
                CALL    WRITE_CH        ; ACTIVE
                POP     ES              ; restore es
                MOV     AX,DX           ; AX=MCB SEGMENT
                INC     AX              ; AX = Segment of next MB
                MOV     ES,AX           ; Also ES
                MOV     AH,49H          ; Free allocated memory function
                INT     21H             ; Actived . .
                JMP     NEXT_MCB        ; Free alloc. next MB

END_ALLOC:
                MOV     ES,DX           ; ES=LAST MCB SEGMENT
                CMP     WORD PTR ES:[1],0  ; IS FREE MB?
                JZ      SET_OLD_VEC     ; YES, NOT FREE ALLOCATE.
                INC     DX
                MOV     ES,DX           ; Next MB
                MOV     AH,49H          ; Free allocated memory function
                INT     21H             ; Actived . .

; ------ Set all interrupt vector to default ( INT 0 - INT 3FH ) ------

SET_OLD_VEC:
                CLI                     ; Clear interrupt
                MOV     CX,200H         ; Word count to be copied
                MOV     SI,OFFSET ALL_VEC  ;
                MOV     AX,CS           ;
                MOV     DS,AX           ; DS:SI = Old interrupt vector
                XOR     DI,DI           ;
                MOV     ES,DI           ; ES:DI = CPU interrupt vector
                REP     MOVSW           ; Copy them

; ------ Restore Peripheral chps Status ----------

                MOV     AL,OLD_8259
                OUT     21H,AL
                MOV     AL,OLD_8255
                OUT     61H,AL

;------ Show Success message ----

                PUSH    CS
                POP     ES
                MOV     SI,OFFSET MESSAGE1      ; Write message
                CALL    WRITE_CH        ; ACTIVED.

                CMP     BYTE PTR CS:KEY_TMP,RM_KEY
                JNZ     EXIT_09

;---- SHOW REMOVED MESSAGE ---

                MOV     SI,OFFSET MESSAGE2
                PUSH    CS
                POP     ES
                CALL    WRITE_CH

;---- REMOVE PKILL.COM ----

                PUSH    CS
                POP     ES
                MOV     AX,ES:[2CH]
                MOV     ES,AX
                MOV     AH,49H
                INT     21H
                PUSH    CS
                POP     ES
                MOV     AH,49H
                INT     21H

;---- SET OLD INT VECTOR NO 9 ----

                MOV     DS,WORD PTR CS:OLD_09_VEC+2
                MOV     DX,WORD PTR CS:OLD_09_VEC
                MOV     AX,2509H
                INT     21H
EXIT_09:
                POP     ES              ;++++++++++++++++++
                POP     DS              ;
                POP     DI              ;
                POP     SI              ;
                POP     DX              ; Restore registers.
                POP     CX              ;
                POP     BX              ;
                POP     AX              ;++++++++++++++++++
                ADD     SP,4            ; Delete old CS:IP
                POPF                    ; Get old flags
                MOV     AH,4CH          ; Terminate the untermination progrram
                INT     21H             ; active
                IRET                    ; End
```

ลิสติงที่ 1 (ต่อ)

```
INT_09H                 ENDP

; WRITE CHARACTER --
; INPUT ES:SI --

WRITE_CH  PROC NEAR

                        PUSH AX
                        PUSH BX
                        MOV  AH,0EH       ; DISPLAY CHARACTER FUNCTION;
                        MOV  BH,CS:VDO_PAGE
NEXT_CH:
                        MOV  AL,BYTE PTR ES:[SI]
                        OR   AL,AL
                        JZ   END_W_CH
                        INT  10H
                        INC  SI
                        JMP  NEXT_CH
END_W_CH:
                        POP  BX
                        POP  AX
                        RET
WRITE_CH    ENDP
; SEARCH PROGRAM NAME --
; INPUT AX=PROGRAM SEGMENT --
; OUTPUT ES:SI = PROGRAM NAME IN ASCIIZ --

SEARCH_PROG PROC NEAR

                          OR      AX,AX                    ; AX=0?
                          JNZ     SEARCH_0                 ; NO, SEARCH PROGRAM
                        MOV     SI,OFFSET  UNKNOWN_MSG     ; SEND BACK
                          PUSH    CS                       ; 'Unknown' MESSAGE
                          POP     ES
SEARCH_0:
                          MOV  ES,AX
                          MOV  SI,ES:[2CH]
                          MOV  ES,SI
                          XOR  SI,SI
NEXT_1:                   CMP  BYTE PTR ES:[SI],0
                          JNZ   NEXT_0
                          CMP  BYTE PTR ES:[SI+1],0
                          JNZ   NEXT_0
                          CMP  BYTE PTR ES:[SI+2],1
                          JNZ   NEXT_0
                          CMP  BYTE PTR ES:[SI+3],0
                          JNZ   NEXT_0
                          ADD  SI,4
                          RET
NEXT_0:                   INC  SI
                          JMP  NEXT_1

SEARCH_PROG ENDP


WAKE_MSG   DB       'Process Killer Woken up !',10,10,13,0
MESSAGE0   DB       'Killed: ',0
MESSAGE1   DB       'Ready . .',10,13,0
MESSAGE2   DB       10,13,'Process Killer Version 2.0  Already Removed.',10,13,0
ENVI_MSG   DB       'Environment of ',0
UNKNOWN_MSG DB      'Unknown',0
CR_LF      DB       0DH,0AH,0
ALL_VEC    DB       1024 DUP (?)                ; Old interrupt vector buffer
                                                ; INT 0H to INT FFH
MARKER2    DW       'XA'                        ; SECOND MARKER
END_RES    LABEL    NEAR                        ; End resident here.

INSTALL:   XOR      AX,AX
           MOV      DX,AX                       ; start at segment 0
NEW_PARA:  MOV      ES,DX                       ; use ES
           CMP      BYTE PTR ES:[0],'M'         ; is MCB?
           JZ       FOUND_MC                    ; yes, feund it
           CMP      BYTE PTR ES:[0],'Z          ; is last mcb?
           JNZ      SKIP_PARA                   ; no, skip
           CMP      WORD PTR ES:[1],0           ; is 0?
           JZ       INSTALL_PROG                ; yes, install prog
SKIP_PARA:
           INC      DX                          ; next segment
           MOV      CX,CS
           DEC      CX
           CMP      DX,CX                       ; check beyond code segment?
           JZ       INSTALL_PROG                ; yes, install program
           JMP      NEW_PARA                    ; no, goto search new paragraph
```

```
FOUND_MCB:
           MOV  AX,ES:[1]
           MOV  ES,AX
           MOV  CX,CS
           CMP  AX,CX                          ; CHECK IF CURRENT CODE
           JZ   INSTALL_PROG                   ; INSTALL PROGRAM
           CMP  WORD PTR ES:[OFFSET MARKER1],'PO'   ; check marker 1
           JNZ  SKIP_PARA
           CMP  WORD PTR ES:[OFFSET MARKER2],'XA'   ; check marker 2
           JNZ  SKIP_PARA
           MOV  DX,OFFSET READY_INST           ; Yes, Display error message
           MOV  AH,9                           ; Display function
           INT  21H                            ; Actived
           MOV  AH,4CH                         ; Terminate function
           INT  21H                            ; Terminate

INSTALL_PROG:
           MOV  AX,3509H                       ; Get INT 9 Vector
           INT  21H                            ; Get it
           MOV  WORD PTR OLD_09_VEC,BX         ; Save offset
           MOV  WORD PTR OLD_09_VEC [2],ES     ; Save segment
           MOV  AX,2509H                       ; Set new INT 9H
           MOV  DX,OFFSET INT_09H              ; to my routine
           INT  21H                            ; Actived

           IN   AL,21H                         ; Load interrupt status
           MOV  OLD_8259,AL                    ; Save it
           IN   AL,61H                         ; Load 8255 status
           MOV  OLD_8255,AL                    ; Save it

           MOV  DI,OFFSET ALL_VEC              ; Old interrupt vector buffer
           MOV  AX,CS
           MOV  ES,AX                          ; ES = CS
           XOR  SI,SI                          ; SI = 0
           MOV  DS,SI                          ; DS:SI = interrupt vector address
           MOV  CX,200H                        ; Word count to be copied
           CLD                                 ; DI,SI Increase
           REP  MOVSW                          ; Copy them

; ------ Get and Save initial video mode ------

           MOV       AX,CS
           MOV       DS,AX                     ; DS = CS
           MOV       AH,0FH                    ; Get video mode function
           INT       10H                       ; Call Bios function
           MOV       VDO_MODE,AL               ; Save current video mode
           MOV       VDO_PAGE,BH               ; Save current video page
           MOV       DX,OFFSET INST_MSG        ; DS:DX = Message to be display
           MOV       AH,9                      ; Display string function
           INT       21H                       ; Actived
           MOV       DX,OFFSET END_RES         ; End resident location
           INT       27H                       ; Terminate and stay resident

INST_MSG   DB        13,10,'Process killer Version 2.0'
           DB        13,10,'Copyright (c) 1990  by Montri Karnjanadecha'
           DB        13,10,10,'Press Ctrl+Alt+RIGHT SHIFT to kill all processes.'
           DB        13,10,10,'Press Ctrl+Alt+RIGHT SHIFT+LEFT SHIFT to remove PKILL.COM'
           DB        13,10,10,'$'
READY_INST DB        13,10,7,'Process killer Version 2.0  already active !'
           DB        13,10  ,'Press Ctrl+Alt+RIGHT SHIFT to activate.'
           DB        13,10,10,'Press Ctrl+Alt+RIGHT SHIFT+LEFT SHIFT to remove PKILL.COM'
           DB        13,10,10,'$'
CODE       ENDS
           END       MAIN
```

หมายเหตุ

TSR = Terminate and Stay Resident Program

เป็นโปรแกรมที่ฝังตัวอยู่ในหน่วยความจำ